Data Quality Implications of Scientific Software Complexity

Julian Newman

Birkbeck College, University of London

juliannewman@ymail.com

Abstract: To be presented at the Seventh Workshop on the Philosophy of Information, "Conceptual challenges of data in science and technology" University College London, 30-31 March 2015

Scientific findings based on computer simulation evoke sceptical responses because data generated by simulation models does not appear to have an objective status comparable with data captured by observation or experiment. Counter to this, philosophers sympathetic to computational science, such as Winsberg (2010) and Humphreys (2004), emphasise parallels between experiment and simulation. Winsberg compares the strategies of simulationists with those of experimenters in augmenting our "reasonable belief" in their results. He extends to simulation Hacking's notion that experimental practice is self-vindicating. Practices such as parameterisation and tricks to overcome computational difficulties "carry with them their own credentials".

Lenhard & Winsberg (2010) argue that complex simulation models face epistemological challenges associated with a novel kind of "confirmation holism". Whereas Duhem argued that no single hypothesis can be tested in isolation, Lenhard & Winsberg claim that it is impossible to locate the sources of the failure of any complex simulation to match known data, because of three characteristics which they regard as intrinsic to the practice of complex systems modelling. These are identified as "fuzzy modularity", "kludging" and "generative entrenchment". Modularity should provide a way of managing complexity; but here the accumulation of more and more interactive sub-models does not break down a complex system into separately manageable pieces, so that it must stand or fall as a whole.

Three related issues are posed by this work: is confirmation holism essential to and unavoidable in complex systems modelling and/or embedded in the specific disciplinary practices of climate science and/or an in-principle remediable failure to observe, recognise and apply available and well-established sound software engineering practices when developing simulation software?

Let us consider the three alleged sources of this novel confirmation holism. "Fuzzy modularity" speaks for itself: the different modules simulate different parts of the climate

system, and these are in continual interaction, such that the behaviour of the sub-models is strongly influenced by the higher level models and through the higher level the sub-models possibly influence one another. Thus it has proved difficult to define clean interfaces between the components of the model. In some respects this problem most readily evokes sympathy in the critic: accepting fuzziness may be seen to enhance the realism of the overall model. But as fuzzy modularity impairs the testability of the software, it can be argued that the price is paid in poor data reliability.

Kludges and Generative Entrenchment are intimately interrelated. A kludge is an inelegant, 'botched together' piece of program, very complex, unprincipled in its design, ill-understood, hard to prove complete or sound and therefore having unknown limitations, and hard to maintain or extend. Generative Entrenchment refers to the historical inheritance of hard-to-alter features from predecessor models: "… complex simulation models in general, and climate models in particular, are – due to fuzzy modularity, kludging and generative entrenchment – the products of their contingent respective histories … As such, climate models are analytically impenetrable in the sense that we have been unable, and are likely to continue to be unable, to attribute the various sources of their successes and failures to their internal modelling assumptions." (Lenhard & Winsberg p 261)

Now let us consider large, complex software systems from the viewpoint of Software Engineering Science, in which the problem of emergent "Architectural Defects" has attracted significant research efforts. A software architecture captures basic design decisions which address qualities central to the system's success, such as performance, reliability, security, maintainability and interoperation. As many as 20% of the defects in a large system can be attributed to architectural decisions, and these can involve twice as much effort to correct as defects arising from mistakes in requirements specification or in the implementation of software components. Li et al (2011) point out that architectural decisions typically affect multiple interacting software components, and as a result architectural defects typically span more than one component: they therefore concentrated on the problems of finding and correcting "multiple-component defects" (MCDs). To this end, they conducted a case study based on the defect records of a large commercial software system which had gone through six releases over a period of 17 years. Compared to single component defects they found that MCDs required more than 20 times as many changes to correct, and that an MCD was 6 to 8 times more likely to persist from one release to another. They identified "architectural hotspots" consisting of 20% of software components in which 80% of MCDs were

concentrated, and these architectural hotspots tended to persist over multiple system releases. This research provides an excellent example of the part played by the "Engineering Science" of Software Engineering in using a large corpus of data to develop a relevant body of theory upon which design and development practice in Software Engineering can build. It exemplifies practices contrary to Humphreys' acceptance of opacity as a sign of the obsolescence of human centred epistemology, or Lenhard & Winsberg's acceptance of confirmation holism with respect to complex system models.

In the practice of software engineering, defects are to be expected, because software development is a human activity which is error-prone, and the more complex the software product the more prone to defects it will be and the harder the defects will be to eliminate. Thus the results of simulation and computational science should never be taken "on trust". Similarly, the fact that a kludge may reflect the model's historical development is in no sense a credential of the model. Kludging in the early stages of a software project creates "Technical Debt" on which interest will accrue in the form of error and maintenance through the lifecycle (Brown et al, 2010; Kruchten et al, 2012). If a simulation is to produce data of sufficient quality to provide guidance for future action, whether in the sphere of engineering or in that of public policy, it must be calibrated using well-attested empirical data. Absent an understanding of the factors that cause a simulation to diverge from past observations (or from empirically-grounded estimates of past trends), the predictive data produced by the simulation have no better standing than mere speculation. Even where the causes of divergence are understood, there is no guarantee that the application of a correction factor will increase the reliability of the predictive outputs; indeed the opposite effect may be produced.

There are growing movements to apply to software the criteria and techniques of rigour that have developed in other sciences, and there is at least some research to suggest that not all climate models are as analytically impenetrable as Lenhard & Winsberg suggest (Pipitone & Easterbrook, 2012). Thus the argument from the essential epistemic opacity of computational science to a non-anthropocentric epistemology runs counter to best practice in software engineering and to empirical results of software engineering science. In software intensive science as elsewhere, data quality depends ultimately upon human scepticism, vigilance and judgment.

REFERENCES

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P et al (2010). Managing technical debt in software-reliant systems. In *Proceedings of the FSE/SDP workshop on Future of software engineering research* (pp. 47-52). ACM.

Kruchten, P., Nord, R. L., & Ozkaya, I. (2012). Technical debt: from metaphor to theory and practice. *IEEE Software*, *29*(6), 18-21.

Lenhard, J & Winsberg, E (2010) "Holism, entrenchment and the future of climate model pluralism." *Studies in the History and Philosophy of Modern Physics,* **41**, 253-263.

Li, Z, Madhavji, NH, Murtaza, SS, Gittens, M, Miranskyy, AV, Godwin, D & Cialini, E (2011) Characteristics of Multiple-component Defects and Architectural Hotspots: A large system case study *Empirical Software Engineering* **16**: 667-702.

Pipitone, J., & Easterbrook, S. (2012). Assessing climate model software quality: a defect density analysis of three models. *Geoscientific Model Development Discussions*, *5*(1), 347-382.

Winsberg, E (2010) *Science in the Age of Computer Simulation.* Chicago University Press.

Submitted 23/01/15

Accepted 09/02/15

Revised 14/02/15